In Proc. of the 2004 Int. Workshop on Description Logics (**DL 2004**)

# *DL-Lite*: Practical Reasoning for Rich DLs

Diego Calvanese[1], Giuseppe De Giacomo[2], Maurizio Lenzerini[2], Riccardo Rosati[2], Guido Vetere[3]

[1] Faculty of Computer Science
Free University of Bolzano/Bozen
Piazza Domenicani 3
I-39100 Bolzano, Italy
calvanese@inf.unibz.it

[2] Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113
I-00198 Roma, Italy
*lastname*@dis.uniroma1.it

[3] IBM Italia
Via Sciangai 53, I-00144 Roma, Italy
guido_vetere@it.ibm.com

**Abstract**

In this paper we study a DL rich enough to express UML class diagrams including ISA and disjointness between classes (but not covering constraints), typing of associations, and participation and functional cardinality constraints. For such a DL, which we call *DL-Lite*, we propose novel reasoning techniques for a variety of tasks, notably including query containment and query answering for conjunctive queries over concepts and roles. The techniques are based on query containment under constraints typical of databases. A prototype implementation of *DL-Lite* has been developed and experimented with.

## 1 Introduction

One of the most important lines of research in Description Logics (DLs) is concerned with the trade-off between expressive power and computational complexity of sound and complete reasoning. Research on this topic has shown that DLs with efficient, i.e., worst-case polynomial time, reasoning algorithms lack modeling power required in capturing conceptual models and basic ontology languages, while DLs with sufficient modeling power suffers from inherently worst-case exponential time behavior of reasoning [8, 9, 2].

In this paper we propose a new DL specifically tailored to capture conceptual data models (e.g., Entity-Relationship) [1], Object-oriented formalisms (e.g., basic UML class diagrams)[1], and basic ontology languages. Notably, we show that advanced forms of sound and complete reasoning, taking into account a knowledge base constituted by a TBox and an ABox, and queries, can be done in polynomial time in the size of the knowledge base. More precisely, our contributions are the following:

1. We define *DL-Lite*, a DL rich enough to capture a significant ontology language. In particular, *DL-Lite* is able to express UML class diagrams including ISA

[1] http://www.omg.org/uml/

and disjointness between classes (but not covering constraints), typing of associations, and cardinality constraints imposing mandatory participation to roles and functionality of roles.

2. For such a DL we propose novel reasoning techniques for a variety of tasks, including conjunctive query answering and containment between conjunctive queries over concepts and roles. Our presentation is focused on the problem of answering conjunctive queries over a knowledge base. We observe that this is one of the few results on answering complex queries (i.e., not corresponding simply to a concept or a role) over a knowledge base [6, 7]. Indeed, answering conjunctive queries over a knowledge base is a challenging problem, even in the case of DL-lite, where the combination of constructs expressible in the knowledge base does not pose particular difficulties in computing subsumption. Our solution bu builds upon and extends a series of techniques developed in databases for query containment and query answering under constraints [10, 3, 4].

3. We show that the above mentioned reasoning tasks can be carried out in polynomial time in the size of the knowledge base.

A prototype implementation of *DL-Lite* has been developed and tested within the SMO (System Management Ontology) project carried out jointly by the University of Rome "La Sapienza" and the IBM Tivoli Laboratory.

The next section defines *DL-Lite* and the associated reasoning services. Section 3 shows that *DL-Lite* is indeed an interesting logic for capturing the basic modeling constructs of conceptual models and ontology languages. Section 4 briefly describes the fundamental reasoning technique associated to *DL-Lite*. Section 5 concludes the paper.

## 2 *DL-Lite*

The DL *DL-Lite* that we present in this paper is quite simple from the language point of view. Namely, starting from atomic concepts, denoted by $A$, possibly with subscripts, and atomic roles, denoted by $R$, possibly with subscripts, we define *basic concepts*, denoted by $B$, as follows:

$$B \ ::= \ A \ \mid \ \exists R \ \mid \ \exists R^-$$

where $A$ is an atomic concept, $\exists R$ is the usual unqualified existentiality on atomic roles $R$, and $\exists R^-$ is the same on *inverse roles*. General concepts in DL-lite are then defined as follows:

$$C \ ::= \ B \ \mid \ \neg B \ \mid \ C_1 \sqcap C_2$$

Note that we have negation on basic concept only and that we have conjunction but not disjunction.

Using this simple language we allow to make assertions of specific forms. Specifically, in a *DL-Lite* TBox, we allow for *inclusion assertions* of the form:

$$B \sqsubseteq C$$

where on the left-hand-side we must have a basic concept ($B$), while on the right-hand-side we may have a general *DL-Lite* concept.

Observe that we do allow for cyclic assertions. Indeed, we can enforce the cyclic propagation of the existence of an $R$-successor using the two *DL-Lite* inclusion assertions $A \sqsubseteq \exists R$, $\exists R^- \sqsubseteq A$. The constraint imposed on a model is similar to the one imposed by the $\mathcal{ALC}$ cyclic assertion $A \sqsubseteq \exists R.\top \sqcap \forall R.A$, though stronger, since it additionally enforces the second component of $R$ to be typed by $A$.

Also, in addition to inclusion assertions, in *DL-Lite* we have *functionality assertions* of the form

$$(\mathsf{funct}\ R), \qquad (\mathsf{funct}\ R^-)$$

expressing, respectively, the functionality of atomic roles and of inverses of atomic roles.

As for the ABox, we allow for membership assertions on atomic concept and on atomic roles:

$$A(a), \qquad R(a,b)$$

stating respectively that the object (denoted by the constant) $a$ is and instance of $A$ and that the pair $(a, b)$ of objects (denoted by the two constants $a$ and $b$) is an instance of the atomic role $R$.

In fact, to denote objects, *DL-Lite* includes two kinds of constants: the usual constants for which the unique name assumption holds, and the so called *soft constants*, which are constants for which the unique name assumption does not hold.

Notice that, using soft constants, we can express in the ABox also membership assertions involving existentials. For example, to express the membership assertion $(\exists R)(a)$, where $a$ is a non-soft constant, we can include in the ABox the membership assertion $R(a, u)$ where $u$ is a fresh (i.e., not used elsewhere in the ABox) soft constant.

Given a *DL-Lite* KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox, we can query the knowledge base using conjunctive queries. A conjunctive query $q$ is an expression of the form

$$\{\, \vec{x} \mid conj(\vec{x}, \vec{y}) \,\}$$

where $\vec{x}$ are the so called distinguished variables that will be bound with object in the KB, $\vec{y}$ are the non-distinguished variables, which are existentially qualified variables, and $conj(\vec{x}, \vec{y})$ is a conjunction of atoms of the form $A(z)$ or $R(z_1, z_2)$ where $A$ and $R$ are respectively atomic concept and roles and $z$, $z_1$, $z_2$ are either constants in the KB or variables in $\vec{x}$ or $\vec{y}$.

The reasoning services that are of interest in *DL-Lite* are:

- *query-answering*: given a query $q(\vec{x})$ with distinguished variables $\vec{x}$ and a knowledge base $\mathcal{K}$, return all tuples $\vec{t}$ of objects that substituted to $\vec{x}$ are such that $\mathcal{K} \models q(\vec{t})$. Observe that as a special case of query answering we have concept satisfiability and logical implication of ABox assertions.

- *query-containment*: given two queries $q_1$ and $q_2$ and a knowledge base $\mathcal{K}$, verify whether $\mathcal{K} \models q_1 \sqsubseteq q_2$, i.e., whether in every model $\mathcal{I}$ of $\mathcal{K}$ the tuples of objects that form the extension of $q_1$ in $\mathcal{I}$ are also in the extension of $q_2$ in $\mathcal{I}$. Observe that as a special case of query containment we have logical implication of inclusion assertions involving atomic concepts on both sides.

In fact, it can be shown that query containment can be reformulated as query answering, in particular with the help of soft constants. Hence, when we discuss reasoning (see Section 4) we will focus on query answering only.

# 3 Why *DL-Lite* is a "rich" DL

Although equipped with advanced reasoning services, at first sight *DL-Lite* seems to be rather weak in modeling intensional knowledge, and hence of limited use in practice. In fact this is not the case. Despite the simplicity of its language and the specific form of inclusion assertions allowed, *DL-Lite* is able to capture the main notions (though not all, obviously) of conceptual modeling formalism used in databases and software engineering such as ER and UML class diagrams.

In particular, *DL-Lite* assertions allow us to specify (below $A$, $A_1$ and $A_2$ are atomic concepts, and $R$ is an atomic role):

- *ISA*, using assertions of the form $A_1 \sqsubseteq A_2$, stating that the class $A_1$ is a subclass of the class $A_2$;
- *class disjointness*, using assertions of the form $A_1 \sqsubseteq \neg A_2$, stating disjointness between the two classes $A_1$ and $A_2$;
- *role-typing*, using assertions of the form $\exists R \sqsubseteq A_1$ (resp., $\exists R^- \sqsubseteq A_2$), stating that the first (resp., second) component of the relation $R$ is of type $A_1$ (resp., $A_2$);
- *participation constraints*, using assertions of the form $A \sqsubseteq \exists R$ (resp., $A \sqsubseteq \exists R^-$), stating that instances of class $A$ participate to the relation $R$ as the first (resp., second) component;
- *non-participation constraints*, using assertions of the form $A \sqsubseteq \neg\exists R$ (resp., $A \sqsubseteq \neg\exists R^-$), stating that instances of class $A$ do not participate to the relation $R$ as the first (resp., second) component;
- *functionality restrictions*, using assertions of the form $(\mathsf{funct}\ R)$ (resp., $(\mathsf{funct}\ R^-)$), stating that an object can be the first (resp., second) component of the relation $R$ at most once.

Notably two important modeling features are missing in *DL-Lite*:

- the ability of stating *covering constraints*, i.e., stating that each instance of a class must be an instance of (at least) one of its subclasses;
- the ability of stating subset constraints between relations.

These features are missing exactly to get the nice computational characteristics that we are after.

Instead, observe that the limitation to binary roles only is not crucial. Indeed, it is possible to extend the reasoning techniques reported here to $n$-ary relations without losing most nice computational properties.

Finally, let us comment on the ability of *DL-Lite* of asserting extensional knowledge using soft constants. These can be considered as an advanced form of *null values* stating that the object with the desired property exists, though its identifier is not known. In other words, soft constants act as existentially quantified variables whose scope is the entire knowledge base.

# 4 Query answering in *DL-Lite*

We now present an algorithm that computes the answers to a conjunctive query over a *DL-Lite* KB. In the following, for ease of exposition we assume that the input query is a boolean query: the extension of the algorithm to non-boolean queries is straightforward.

Due to space limitations, we are only able to provide an informal description of the algorithm; moreover, we assume that no soft constants are present in the ABox.

## 4.1 Algorithm

The algorithm takes as input a *DL-Lite* KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ and a boolean conjunctive query $q$, and returns a boolean value. The algorithm consists of five steps:

1. *TBox normalization*: inclusion assertions of $\mathcal{T}$ in which conjunctive concepts occur in the right-hand side are rewritten by iterative application of the rule: if $B \sqsubseteq C_1 \sqcap C_2$ occurs in $\mathcal{T}$, then replace this assertion in $\mathcal{T}$ with the two assertions $B \sqsubseteq C_1$ , $B \sqsubseteq C_2$. The normalized TBox resulting from such a transformation contains the following types of assertions:
   - *ISA assertions* of the form $A_1 \sqsubseteq A_2$, where $A_1$ and $A_2$ are atomic concepts;
   - *disjointness assertions* of the form $B_1 \sqsubseteq \neg B_2$ where $B_1$ is a basic concept (i.e., either an atomic or an existential concept) and $\neg B_2$ is a negated basic concept;
   - *role-typing assertions* of the form $\exists R \sqsubseteq B$ or $\exists R^- \sqsubseteq B$, where $B$ is a basic concept;
   - *participation assertions* of the form $A \sqsubseteq \exists R$ or $A \sqsubseteq \exists R^-$, where $A$ is an atomic concept;
   - *functionality assertions* of the form $(\mathsf{funct}\ R)$ or $(\mathsf{funct}\ R^-)$.
2. *KB consistency check*: this step checks whether the ABox $\mathcal{A}$ satisfies the functionality and disjointness assertions occurring in the TBox $\mathcal{T}$. Specifically:

(a) First, in order to check satisfiability w.r.t. disjointness assertions, the TBox is expanded by computing all the disjointness assertions implied by the inclusion assertions in $\mathcal{T}$. More precisely, the TBox $\mathcal{T}$ is closed with respect to the following inference rule: if the assertions $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_3$ occur in $\mathcal{T}$ (where $C_1, C_2, C_3$ are arbitrary concepts), then add the assertion $C_1 \sqsubseteq C_3$ to $\mathcal{T}$.

(b) Then, the algorithm checks satisfiability w.r.t. disjointness assertions in $\mathcal{T}$: e.g., the assertion $B_1 \sqsubseteq \neg B_2$ in $\mathcal{T}$ is satisfied in $\mathcal{A}$ iff there is no $a$ such that both $a : B_1$ and $a : B_2$ are in $\mathcal{A}$ (if $B_2$ is the existential concept $\exists R$ (resp., $\exists R^-$), then also assertions of the form $R(a, b)$ (resp., $R(b, a)$) are taken into account).

(c) Finally, also satisfiability of $\mathcal{A}$ w.r.t. functionality assertions is checked: e.g., the assertion ($\mathsf{funct}$ $R$) in $\mathcal{T}$ is satisfied in $\mathcal{A}$ iff there is no pair of assertions in $\mathcal{A}$ of the form $R(a, b)$, $R(a, c)$.

If there is a disjointness assertion or a functionality assertion in $\mathcal{T}$ that the ABox $\mathcal{A}$ does not satisfy, then the algorithm returns true (there is no model for the KB $\mathcal{K}$, therefore every query is trivially true), otherwise the algorithm executes the next step.

3. *Query expansion*: the conjunctive query is rewritten based on the ISA, role-typing, and participation assertions in $\mathcal{T}$. More specifically, starting from the initial conjunctive query, a union of conjunctive queries is computed, by essentially applying the ISA, role-typing, and participation assertions as concept rewriting rules, applied from right to left. For instance, in the presence of the ISA assertion $A \sqsubseteq C$, the query $C(a)$ can be rewritten as $A(a)$, while in the presence of the role-typing assertion $\exists R \sqsubseteq C$, the same query can be rewritten as $R(a, x)$, where $x$ is a new variable symbol. Intuitively, in expanding the query we essentially embed all the relevant knowledge of the TBox represented by ISA, role-typing, and participation assertions.

4. *Query evaluation*: Finally, the expanded query is evaluated over the ABox $\mathcal{A}$. More precisely, the algorithm returns true if and only if there exists a conjunct of the expanded query that has an image in the ABox. Basically, a conjunct has an image in the ABox $\mathcal{A}$ if there exists a substitution $\sigma$ from the variables occurring in the conjunct to the constants occurring in $\mathcal{A}$ such that for each atom $\phi$ occurring in the conjunct, $\sigma(\phi) \in \mathcal{A}$ (actually, if an existential concept occurs in the atom, then also role memberhip assertions can provide an image for the atom). In other words, the algorithm evaluates the union of conjunctive queries considering the ABox as a database.

## 4.2 Correctness

Informally, the correctness of the above reasoning technique is essentially due to the fact that the assertions in the TBox can be divided into two classes:

- disjointness and functionality assertions, taken into account by Step 2 of the algorithm;
- ISA, role-typing, and participation assertions, considered in Step 3.

Indeed, it can be shown that the interaction between these two classes of assertions is limited to the derivation of new disjointness assertions in the TBox closure computed during Step 2. After these steps, the TBox can be discarded in the final query evaluation step.

### 4.3 Complexity

As for the complexity of the algorithm, it is easy to prove that the algorithm runs in polynomial time with respect to the size of the knowledge base $\mathcal{K}$, while the computation time is exponential with respect to the size of the query. The latter is due to the fact that the union of conjunctive queries computed in Step 3 may consist of a number of disjuncts (each of polynomial size) that is exponential in the number of atoms in the body of the initial query. Moreover, the evaluation of each disjunct in Step 4 may take nondeterministic polynomial (i.e., for practical purposes, exponential) time in the number of atoms of the disjunct, and hence in the number of atoms in the body of the initial query.

The algorithm can be extended to the presence of soft constants in the ABox, by adding a unification step that takes into account the presence of functionality assertions on the soft constants. Such an extension does not affect the computational properties of the algorithm.

Finally, it can be shown that, in the presence of soft constants, containment between conjunctive queries can be immediately reduced to query answering by the well-known "query freezing" technique (see, e.g., [11]), in which soft constants are used to deal with possible equalities implied by functionality assertions.

Summarizing, the following property holds.

**Theorem 1** *Subsumption, query answering, and query containment in DL-Lite are polynomial in the size of the knowledge base.*

## 5 Conclusions

We have described *DL-Lite*, a new DL specifically tailored to capture conceptual data models and basic ontology languages, while keeping the worst-case complexity of sound and complete reasoning tractable.

In this paper we focused on binary roles only, but this is not a crucial limitation. Indeed, it is possible to extend the reasoning techniques reported here to $n$-ary relations without loosing their nice computational properties. On the other hand, the results reported in [5] imply that the introduction of subset constraints on roles (i.e., role inclusion assertions) makes our technique inapplicable.

**Acknowledgments** This research was partly supported by MIUR under FIRB (Fondo per gli Investimenti della Ricerca di Base) project "MAIS: Multichannel Adaptive Information Systems" in the context of the Workpackage 2 activities, and by the EU funded projects INFOMIX (IST-2001-33570) and SEWASIE (IST-2001-34825).

## References

[1] C. Batini, S. Ceri, and S. B. Navathe. *Conceptual Database Design, an Entity-Relationship Approach.* Benjamin and Cummings Publ. Co., Menlo Park, California, 1992.

[2] A. Borgida and R. J. Brachman. Conceptual modeling with description logics. In F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors, *The Description Logic Handbook: Theory, Implementation and Applications*, chapter 10, pages 349–372. Cambridge University Press, 2003.

[3] A. Calì, D. Lembo, and R. Rosati. On the decidability and complexity of query answering over inconsistent and incomplete databases. In *Proc. of the 22nd ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS 2003)*, pages 260–271, 2003.

[4] A. Calì, D. Lembo, and R. Rosati. Query rewriting and answering under constraints in data integration systems. In *Proc. of the 18th Int. Joint Conf. on Artificial Intelligence (IJCAI 2003)*, pages 16–21, 2003.

[5] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. What to ask to a peer: Ontology-based query reformulation. In *Proc. of the 9th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2004)*, 2004.

[6] D. Calvanese, G. De Giacomo, and M. Lenzerini. On the decidability of query containment under constraints. In *Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'98)*, pages 149–158, 1998.

[7] D. Calvanese, G. De Giacomo, and M. Lenzerini. Answering queries using views over description logics knowledge bases. In *Proc. of the 17th Nat. Conf. on Artificial Intelligence (AAAI 2000)*, pages 386–391, 2000.

[8] D. Calvanese, M. Lenzerini, and D. Nardi. Description logics for conceptual data modeling. In J. Chomicki and G. Saake, editors, *Logics for Databases and Information Systems*, pages 229–264. Kluwer Academic Publisher, 1998.

[9] D. Calvanese, M. Lenzerini, and D. Nardi. Unifying class-based representation formalisms. *J. of Artificial Intelligence Research*, 11:199–240, 1999.

[10] D. S. Johnson and A. C. Klug. Testing containment of conjunctive queries under functional and inclusion dependencies. *J. of Computer and System Sciences*, 28(1):167–189, 1984.

[11] J. D. Ullman. Information integration using logical views. In *Proc. of the 6th Int. Conf. on Database Theory (ICDT'97)*, volume 1186 of *Lecture Notes in Computer Science*, pages 19–40. Springer, 1997.